

# Linear Parameter Varying Iterative Learning Control

Mark Butcher and Alireza Karimi

**Abstract**—In this paper an Iterative Learning Control (ILC) algorithm is proposed for a certain class of Linear Parameter Varying (LPV) systems whose dynamics change between iterations. Consistency of the algorithm in the presence of stochastic disturbances is shown. The proposed algorithm is tested in simulation and the obtained tracking performance is compared with that obtained using a standard Linear Time Invariant ILC algorithm. Better results are obtained using the proposed method.

**Index Terms**—Learning control systems, linear parameter-varying system, iterative learning control

## I. INTRODUCTION

Iterative Learning Control (ILC) is now well recognised as a methodology capable of producing very high tracking performance for systems carrying out repetitive tasks [1], [2]. ILC adjusts the system's input from one repetition/iteration to the next in order to compensate for the system's dynamics and disturbances, thus improving its output tracking performance. ILC has been successfully applied to numerous systems carrying out repetitive tasks (e.g. [3], [4], [5], [6]). One of the main drawbacks of ILC, however, is that it requires the system's dynamics and disturbances to be repetition invariant. This assumption may not always be valid in practice i.e. when the output sensor is affected by non-negligible measurement noise or if a robotic arm picks up different loads from one repetition to the next. If the variations are of a stochastic nature, they should not be compensated for and ILC algorithms that are insensitive to these variations should be used (e.g. [7]). If, however, they are deterministic, it may be possible to compensate for the changes. It is therefore of interest to investigate ILC algorithms that are applicable when the system's dynamics and disturbances change deterministically between repetitions.

To the authors' knowledge, very little work has been done on this problem. In [8] the problem of deterministically iteration varying disturbances is considered. It is shown that, by using the internal-model principle in the iteration domain, the disturbances can be rejected as the iterations tend to infinity. The problem with this approach is that it is necessary to know the form of the disturbance variation in advance in order to include its model in the ILC controller, as required by the internal-model principle.

In this paper we develop an ILC algorithm that can lead to improved tracking for systems that can be represented by the Linear Parameter Varying (LPV) class of systems

[9], and therefore whose dynamics change as a function of a measurable scheduling parameter. Development and application of control techniques for LPV systems has been active in recent years (e.g. [10], [11]). ILC for LPV systems has been considered in [12]. The variation of the system's dynamics due to the changing scheduling parameter is, however, assumed to take place during the iteration, rather than from one iteration to the next. The problem considered is, therefore, different to that studied here.

The paper is organised as follows. In Section II the LPV ILC algorithm is presented. The algorithm is then tested in simulation in Section III. Finally some conclusions are made in Section IV.

## II. LINEAR PARAMETER VARYING ILC

In this section the LPV ILC algorithm is described. First the class of systems considered in this paper is outlined. The ideal input for this class of systems is then given. A parameterisation for the input to be estimated is then presented, based on the structure of the ideal input. Next a recursive algorithm, similar to the standard recursive least squares but taking into account the iteration-varying aspect of LPV systems, is developed to estimate the ideal input. Finally it is shown that, under a persistency of excitation condition on the scheduling parameter, the learning algorithm converges probabilistically to the ideal input.

### A. System description

We consider the finite-time tracking problem of following a repetitive, finite duration desired trajectory  $y_d(t)$ , defined for  $t = 0, \dots, N - 1$ .

The output at time  $t$  of the stable, LPV SISO discrete-time system, resulting from linearising a nonlinear system about the operating point  $\sigma(k) \in \mathbb{R}^{n_\sigma}$ , is given by:

$$\begin{aligned} A(\sigma(k), q^{-1})y(t, k, \sigma(k)) \\ = B(q^{-1})u(t, k) + d(t, k, \sigma(k)) + n(t, k, \sigma(k)), \end{aligned} \quad (1)$$

where

$$A(\sigma(k), q^{-1}) = \sum_{j=0}^{n_a} a_j(\sigma(k))q^{-j}, \quad B(q^{-1}) = \sum_{j=0}^{n_b} b_j q^{-j},$$

$u(t, k)$  is the input to the system and  $q^{-1}$  is the backward-shift time operator.  $d(t, k, \sigma(k))$  and  $n(t, k, \sigma(k))$  are a deterministic and a stochastic disturbance, respectively, both possibly dependent on  $\sigma$ . The operating point  $\sigma(k)$  remains constant throughout repetition  $k$ . The dependence of the

The authors are with the Automatic Control Laboratory, Ecole Polytechnique Fédérale de Lausanne (EPFL), 1015 Lausanne, Switzerland.

Corresponding author: alireza.karimi@epfl.ch

This work is supported by the Swiss National Science Foundation under Grant No. 200021-116156/1.

coefficients  $a_i$  and the deterministic disturbance on the scheduling parameter is assumed polytopic:

$$\begin{aligned} a_i(\boldsymbol{\sigma}(k)) &= \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) a_{i,j} \\ d(t, k, \boldsymbol{\sigma}(k)) &= \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) d_j(t), \\ 0 \leq \lambda_j(\boldsymbol{\sigma}(k)) &\leq 1, \quad \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) = 1, \end{aligned} \quad (2)$$

where  $\lambda_j(\boldsymbol{\sigma}(k)) : \mathbb{R}^{n_\sigma} \rightarrow \mathbb{R}$ . Additionally we designate the values of  $\boldsymbol{\sigma}$  at the vertices of the polytopic space as  $\boldsymbol{\sigma}_j$ .

As the signals are defined over the finite duration of the repetition, it is possible to express the system's input-output relationship by a matrix representation. Using the lifted-system representation typically used in ILC, we define, for a system with a relative degree of  $m$ , the vectors:

$$\begin{aligned} \mathbf{u}(k) &= [u(0, k), u(1, k), \dots, u(N - m - 1, k)]^T \\ \mathbf{y}(k, \boldsymbol{\sigma}(k)) &= [y(m, k, \boldsymbol{\sigma}(k)), y(m + 1, k, \boldsymbol{\sigma}(k)), \\ &\quad \dots, y(N - 1, k, \boldsymbol{\sigma}(k))]^T, \end{aligned} \quad (3)$$

with  $\mathbf{y}_d$ ,  $\mathbf{d}_j$  and  $\mathbf{n}(k, \boldsymbol{\sigma}(k))$  defined similarly to  $\mathbf{y}(k, \boldsymbol{\sigma}(k))$ . This representation can then be used to write (1) as:

$$\begin{aligned} \mathbf{A}(\boldsymbol{\sigma}(k)) \mathbf{y}(k, \boldsymbol{\sigma}(k)) &= \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) \mathbf{A}_j \mathbf{y}(k, \boldsymbol{\sigma}(k)) \\ &= \mathbf{B} \mathbf{u}(k) + \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) \mathbf{d}_j + \mathbf{n}(k, \boldsymbol{\sigma}(k)), \end{aligned} \quad (4)$$

$$\begin{aligned} \text{where } \mathbf{A}_j &= \begin{bmatrix} a_{0,j} & 0 & \dots & 0 \\ a_{1,j} & a_{0,j} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{N-m-1,j} & a_{N-m-2,j} & \dots & a_{0,j} \end{bmatrix} \\ \text{and } \mathbf{B} &= \begin{bmatrix} b_m & 0 & \dots & 0 \\ b_{m+1} & b_m & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ b_{N-1} & b_{N-2} & \dots & b_m \end{bmatrix}. \end{aligned} \quad (5)$$

*Assumptions:* The disturbance vector  $\mathbf{n}(k, \boldsymbol{\sigma}(k))$  is assumed to be a zero-mean, random vector with unknown but bounded covariance matrix  $\mathbf{R}_n(k, \boldsymbol{\sigma}(k))$ . Additionally, realisations of  $\mathbf{n}(k, \boldsymbol{\sigma}(k))$  are considered independent.

Later in the paper the following lemma will be needed:

*Lemma 1:* [p. 253 in [13]] Let  $X(k)$  be an independent random sequence with constant mean  $\mu_X$  and variance  $\sigma_X^2(k)$  defined for  $k \geq 1$ . Define another random sequence as:

$$\hat{\mu}_X(K) = \frac{1}{K} \sum_{k=1}^K X(k) \quad \text{for } K \geq 1. \quad (6)$$

Then if

$$\lim_{K \rightarrow \infty} \sum_{k=1}^K \frac{\sigma_X^2(k)}{k^2} < \infty, \quad (7)$$

$$\hat{\mu}_X(K) \xrightarrow{P} \mu_X \text{ as } K \rightarrow \infty, \quad (8)$$

where  $\xrightarrow{P}$  represents convergence in probability.

The theorem also holds true for almost sure (a.s.) convergence, making it a Strong Law.

### B. Ideal input

The measured error is defined as:

$$\mathbf{e}(k, \boldsymbol{\sigma}(k)) = \mathbf{y}_d - \mathbf{y}(k, \boldsymbol{\sigma}(k)). \quad (9)$$

The ideal input vector, defined as the one that achieves zero mean error, i.e.  $E\{\mathbf{e}(k, \boldsymbol{\sigma}(k))\} = \mathbf{0} \forall \boldsymbol{\sigma}$ , where  $E\{\cdot\}$  denotes the mathematical expectation operator, is given by:

$$\mathbf{u}^0(k, \boldsymbol{\sigma}(k)) = \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) \mathbf{B}^{-1} (\mathbf{A}_j \mathbf{y}_d - \mathbf{d}_j). \quad (10)$$

At the values of  $\boldsymbol{\sigma}$  that correspond to the vertices of the polytopic dependence we have:

$$\mathbf{u}^0(k, \boldsymbol{\sigma}_j) = \mathbf{B}^{-1} (\mathbf{A}_j \mathbf{y}_d - \mathbf{d}_j). \quad (11)$$

Using this (10) can be written as:

$$\mathbf{u}^0(k, \boldsymbol{\sigma}(k)) = \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) \mathbf{u}^0(\boldsymbol{\sigma}_j). \quad (12)$$

*Remark:* Expression (12) rationalises the class of LPV systems considered here i.e. LPV systems with dependence on the scheduling parameter solely in the denominator. Only for this system class does the ideal input depend linearly on the ideal inputs at the vertices. This linear dependence means the estimation of these inputs, as considered in the next subsection, can be done via linear least squares and there will be a global minimum. Furthermore, as will be seen in the application section, real systems exist that belong to this system class.

### C. Input parameterisation

We see from (12) that the ideal input is a function of the scheduling parameter  $\boldsymbol{\sigma}(k)$ . The ILC algorithm should therefore estimate an input that is also a function of  $\boldsymbol{\sigma}(k)$ . Motivated by the form of the ideal input, and under the assumption that the functions  $\lambda_j(\boldsymbol{\sigma}(k))$  are known, the input is parameterised as:

$$\mathbf{u}(k, \boldsymbol{\sigma}(k)) = \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) \mathbf{u}_j. \quad (13)$$

The system output can be written as:

$$\begin{aligned} \mathbf{y}(k, \boldsymbol{\sigma}(k)) &= \mathbf{A}^{-1}(\boldsymbol{\sigma}(k)) [\mathbf{B} \mathbf{u}(k) + \mathbf{d}(k, \boldsymbol{\sigma}(k)) \\ &\quad + \mathbf{n}(k, \boldsymbol{\sigma}(k))] \\ &= \mathbf{G}(\boldsymbol{\sigma}(k)) \mathbf{u}(k) + \bar{\mathbf{d}}(k, \boldsymbol{\sigma}(k)) + \bar{\mathbf{n}}(k, \boldsymbol{\sigma}(k)), \end{aligned} \quad (14)$$

where

$$\mathbf{G}(\boldsymbol{\sigma}(k)) = \mathbf{A}^{-1}(\boldsymbol{\sigma}(k))\mathbf{B}$$

$$\bar{\mathbf{d}}(k, \boldsymbol{\sigma}(k)) = \mathbf{A}^{-1}(\boldsymbol{\sigma}(k))\mathbf{d}(k, \boldsymbol{\sigma}(k))$$

and

$$\bar{\mathbf{n}}(k, \boldsymbol{\sigma}(k)) = \mathbf{A}^{-1}(\boldsymbol{\sigma}(k))\mathbf{n}(k, \boldsymbol{\sigma}(k)).$$

It should be mentioned that for  $\mathbf{G}(\boldsymbol{\sigma}(k))$ ,  $\bar{\mathbf{d}}(k, \boldsymbol{\sigma}(k))$  and  $\bar{\mathbf{n}}(k, \boldsymbol{\sigma}(k))$  to exist,  $\mathbf{A}(\boldsymbol{\sigma}(k))$  should be nonsingular, which is always the case due to its structure. If the parameterised input (13) is applied to the system we have:

$$\begin{aligned} \mathbf{y}(k, \boldsymbol{\sigma}(k)) &= \mathbf{G}(\boldsymbol{\sigma}(k)) \sum_{j=0}^{J-1} \lambda_j(\boldsymbol{\sigma}(k)) \mathbf{u}_j + \bar{\mathbf{d}}(k, \boldsymbol{\sigma}(k)) \\ &\quad + \bar{\mathbf{n}}(k, \boldsymbol{\sigma}(k)) \\ &= [\lambda_0(\boldsymbol{\sigma}(k))\mathbf{G}(\boldsymbol{\sigma}(k)), \lambda_1(\boldsymbol{\sigma}(k))\mathbf{G}(\boldsymbol{\sigma}(k)), \dots, \\ &\quad \lambda_{J-1}(\boldsymbol{\sigma}(k))\mathbf{G}(\boldsymbol{\sigma}(k))] [\mathbf{u}_0^T, \mathbf{u}_1^T, \dots, \mathbf{u}_{J-1}^T]^T \\ &\quad + \bar{\mathbf{d}}(k, \boldsymbol{\sigma}(k)) + \bar{\mathbf{n}}(k, \boldsymbol{\sigma}(k)) \\ &= \mathcal{G}(\boldsymbol{\sigma}(k))\mathbf{U} + \bar{\mathbf{d}}(k, \boldsymbol{\sigma}(k)) + \bar{\mathbf{n}}(k, \boldsymbol{\sigma}(k)), \end{aligned} \quad (15)$$

where  $\mathcal{G}(\boldsymbol{\sigma}(k)) \in \mathbb{R}^{(N-m) \times J(N-m)}$  and  $\mathbf{U} \in \mathbb{R}^{J(N-m)}$ .

The ideal input (12) is achieved when:

$$\mathbf{U} = \mathbf{U}^0 = [[\mathbf{u}^0(\boldsymbol{\sigma}_0)]^T, [\mathbf{u}^0(\boldsymbol{\sigma}_1)]^T, \dots, [\mathbf{u}^0(\boldsymbol{\sigma}_{J-1})]^T]^T. \quad (16)$$

#### D. The learning algorithm

The aim of the algorithm is to estimate  $\mathbf{U}^0$  over the iterations. The approach proposed here to find the estimate is to minimise a quadratic cost function over all previous iterations i.e. to find the estimate that minimises:

$$J_K(\mathbf{U}) = \frac{1}{2K} \sum_{k=1}^K \mathbf{e}^T(k, \boldsymbol{\sigma}(k), \mathbf{U}) \mathbf{e}(k, \boldsymbol{\sigma}(k), \mathbf{U}), \quad (17)$$

where  $K$  is the number of completed iterations. Via some simple calculations, the estimate can be found as:

$$\hat{\mathbf{U}}_K = \mathbf{P}(K) \sum_{k=1}^K \mathcal{G}^T(\boldsymbol{\sigma}(k)) [\mathbf{y}_d - \bar{\mathbf{d}}(k, \boldsymbol{\sigma}(k)) - \bar{\mathbf{n}}(k, \boldsymbol{\sigma}(k))], \quad (18)$$

where

$$\mathbf{P}(K) = \left[ \sum_{k=1}^K \mathcal{G}^T(\boldsymbol{\sigma}(k)) \mathcal{G}(\boldsymbol{\sigma}(k)) \right]^{-1}.$$

Alternatively, via some standard manipulations, (18) can be written in the recursive form as:

$$\begin{aligned} \hat{\mathbf{U}}_{k+1} &= \hat{\mathbf{U}}_k \\ &\quad + \mathbf{P}(k+1) \mathcal{G}^T(\boldsymbol{\sigma}(k+1)) \mathbf{e}(k+1, \boldsymbol{\sigma}(k+1), \hat{\mathbf{U}}_k). \end{aligned} \quad (19)$$

The error signal:

$$\begin{aligned} \mathbf{e}(k+1, \boldsymbol{\sigma}(k+1), \hat{\mathbf{U}}_k) &= \mathbf{y}_d - \mathcal{G}(\boldsymbol{\sigma}(k+1)) \hat{\mathbf{U}}_k \\ &\quad - \bar{\mathbf{d}}(k+1, \boldsymbol{\sigma}(k+1)) - \bar{\mathbf{n}}(k+1, \boldsymbol{\sigma}(k+1)) \end{aligned}$$

can be evaluated experimentally by applying the input  $\mathbf{u}(k, \boldsymbol{\sigma}(k+1))$  from (13) based on  $\hat{\mathbf{U}}_k$  to the real system. Therefore we see that the estimate  $\hat{\mathbf{U}}$  can be evaluated recursively using data measured from the real system.

#### E. Consistency of estimates

Next a condition for consistent estimates is given.

*Theorem 1:* Under the assumptions made in Subsection II-A, the algorithm (19) is a consistent estimator, i.e.  $\hat{\mathbf{U}}_K$  converges almost surely to  $\mathbf{U}^0$  as  $K \rightarrow \infty$ , if:

$$\lim_{K \rightarrow \infty} \frac{1}{K} \mathbf{P}^{-1}(K) \quad (20)$$

is nonsingular.

*Proof:* The recursive algorithm (19) has the same asymptotic properties as the batch result (18) so the consistency of (18) can be considered. (18) can be rewritten as:

$$\begin{aligned} \hat{\mathbf{U}}_K &= K \mathbf{P}(K) \\ &\quad \frac{1}{K} \sum_{k=1}^K \mathcal{G}^T(\boldsymbol{\sigma}(k)) [\mathcal{G}(\boldsymbol{\sigma}(k)) \mathbf{U}^0 - \bar{\mathbf{n}}(k, \boldsymbol{\sigma}(k))] \\ &= \mathbf{U}^0 - K \mathbf{P}(K) \frac{1}{K} \sum_{k=1}^K \mathcal{G}^T(\boldsymbol{\sigma}(k)) \bar{\mathbf{n}}(k, \boldsymbol{\sigma}(k)). \end{aligned}$$

In order for the estimates to be consistent it is necessary that:

$$\lim_{K \rightarrow \infty} \frac{1}{K} \mathbf{P}^{-1}(K) \quad (21)$$

be nonsingular and

$$\begin{aligned} \mathbf{w}(K) &= \frac{1}{K} \sum_{k=1}^K \mathcal{G}^T(\boldsymbol{\sigma}(k)) \bar{\mathbf{n}}(k, \boldsymbol{\sigma}(k)) \rightarrow \mathbf{0} \\ &\quad \text{a.s., as } K \rightarrow \infty. \end{aligned} \quad (22)$$

(21) is the condition in the theorem that should be satisfied. To show that (22) is true we first write:

$$\begin{aligned} \mathcal{G}^T(\boldsymbol{\sigma}(k)) \bar{\mathbf{n}}(k, \boldsymbol{\sigma}(k)) &= \mathcal{G}^T(\boldsymbol{\sigma}(k)) \mathbf{A}^{-1}(\boldsymbol{\sigma}(k)) \mathbf{n}(k, \boldsymbol{\sigma}(k)) \\ &= \mathbf{x}(k, \boldsymbol{\sigma}(k)). \end{aligned}$$

Since  $\mathbf{n}(k, \boldsymbol{\sigma}(k))$  is assumed to be zero mean and independent between iterations, and  $\mathbf{A}$  is nonsingular, this expression means that  $\mathbf{x}(k, \boldsymbol{\sigma}(k))$  will also be zero mean and independent between iterations. Additionally  $\mathbf{x}(k, \boldsymbol{\sigma}(k))$ 's covariance matrix is given by:

$$\begin{aligned} \mathbf{R}_{\mathbf{x}}(k, \boldsymbol{\sigma}(k)) &= \mathcal{G}^T(\boldsymbol{\sigma}(k)) \mathbf{A}^{-1}(\boldsymbol{\sigma}(k)) \mathbf{R}_{\mathbf{n}}(k, \boldsymbol{\sigma}(k)) \mathbf{A}^{-T}(\boldsymbol{\sigma}(k)) \mathcal{G}(\boldsymbol{\sigma}(k)). \end{aligned}$$

Since  $\mathbf{R}_{\mathbf{n}}(k, \boldsymbol{\sigma}(k))$  is assumed bounded and  $\mathbf{A}$  is nonsingular,  $\mathbf{R}_{\mathbf{x}}(k, \boldsymbol{\sigma}(k))$  will be bounded.

The  $i$ th component of  $\mathbf{w}(K)$  in (22) therefore represents the sample average of a sequence of zero-mean, independent random variables with finite, though possibly different, variances  $\sigma_x^2(i, k, \boldsymbol{\sigma}(k))$ . Lemma 1 implies that (22) is satisfied if:

$$\lim_{K \rightarrow \infty} \sum_{k=1}^K \frac{\sigma_x^2(i, k, \boldsymbol{\sigma}(k))}{k^2} < \infty. \quad (23)$$

Since

$$\sum_{k=1}^K \frac{\sigma_x^2(i, k, \sigma(k))}{k^2} \leq \bar{\sigma}_x^2(i) \sum_{k=1}^K \frac{1}{k^2}$$

where  $\sigma_x^2(i, k, \sigma(k)) \leq \bar{\sigma}_x^2(i) < \infty \forall k$ , and

$$\lim_{K \rightarrow \infty} \sum_{k=1}^K \frac{1}{k^2} = \frac{\pi}{6},$$

(23) is satisfied. The theorem is therefore proved.  $\blacksquare$

*Remark:* The condition (20) in Theorem 1 is a persistency of excitation condition that requires the scheduling parameter trajectory to be sufficiently rich.

### III. SIMULATION

The effectiveness of the proposed method is next shown via simulation. The LPV system used in the simulation is defined by the polynomials:

$$B(q^{-1}) = (0.0048 + 0.0047q^{-1})q^{-1} \quad (24)$$

and

$$A(\sigma(k), q^{-1}) = 1 + a_1(\sigma(k))q^{-1} + a_2(\sigma(k))q^{-2} \quad (25)$$

where

$$a_1(\sigma(k)) = -1.8953\lambda_0(\sigma(k)) - 1.8903\lambda_1(\sigma(k)) \quad (26)$$

and

$$a_2(\sigma(k)) = 0.9048\lambda_0(\sigma(k)) + 0.9098\lambda_1(\sigma(k)). \quad (27)$$

The functions  $\lambda_j(\sigma(k))$  are given by:

$$\lambda_0(\sigma(k)) = \frac{\bar{\sigma} - \sigma(k)}{\bar{\sigma} - \underline{\sigma}}, \quad \lambda_1(\sigma(k)) = \frac{\sigma(k) - \underline{\sigma}}{\bar{\sigma} - \underline{\sigma}},$$

where  $\underline{\sigma}$  and  $\bar{\sigma}$  are the respective minimum and maximum values of the scheduling parameter  $\sigma(k)$ , which is taken as a uniformly distributed signal in the interval (0,1). We, therefore, have that  $\underline{\sigma} = 0$  and  $\bar{\sigma} = 1$ .

The desired output  $y_d(t)$  is defined by:

$$y_d(t) = 1 - \cos(0.01\pi t) \quad t = 0, 1, \dots, 200. \quad (28)$$

The disturbances  $d(t, k, \sigma(k))$  and  $n(t, k, \sigma(k))$  are set to zero to emphasise the proposed algorithm's ability to compensate the changing dynamics.

The proposed method is applied to the system, however, in order to investigate the method's robustness to uncertainty, instead of using the matrix  $\mathbf{G}(\sigma(k))$  in the algorithm, the constant matrix  $\mathbf{G}(\underline{\sigma})$  is used, independently of the current value of  $\sigma(k)$ .

In the implementation of the algorithm the matrix:

$$\mathbf{P}(k+1) = \left[ \sum_{i=1}^{k+1} \mathcal{G}^T(\sigma(k)) \mathcal{G}(\sigma(k)) \right]^{-1}$$

where

$$\mathcal{G}(\sigma(k)) = [\lambda_0(\sigma(k))\mathbf{G}(\underline{\sigma}), \lambda_1(\sigma(k))\mathbf{G}(\underline{\sigma})]$$

is required at each iteration of the algorithm. In order for the inverse to exist, it is necessary that this matrix be full rank.

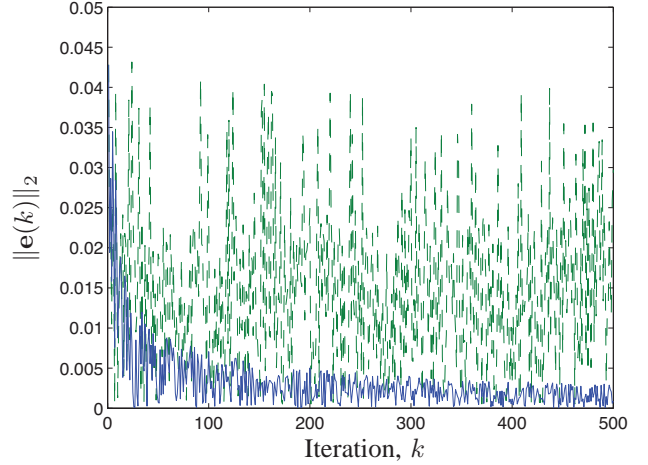


Fig. 1. RMS value of  $\mathbf{e}(k)$  obtained in simulation using the proposed algorithm (blue) and the LTI algorithm (green-dashed)

For this to be the case a persistency of excitation condition on the scheduling parameter trajectory should be satisfied. A necessary, but not sufficient, condition for the matrix to be full rank is that 2 different values of  $\sigma(k)$  are visited. This condition arises because  $\mathcal{G}(\sigma(k)) \in \mathbb{R}^{(N-m) \times 2(N-m)}$  is of maximum rank  $N - m$ , and thus so is  $\mathcal{G}^T(\sigma(k))\mathcal{G}(\sigma(k))$ .  $\mathbf{P}^{-1}(k+1)$  must be of rank  $2(N - m)$ . This is only possible after 2 different values of  $\sigma(k)$  are visited. In order to satisfy this the algorithm is not used to calculate the system input until 2 different values of  $\sigma(k)$  have been visited.  $\mathbf{u}(k, \sigma(k)) = \mathbf{y}_d$  is used for  $k = 1, 2$ .

The results obtained using the proposed method are compared with those obtained using an ILC algorithm developed under the assumption of the system being LTI. The algorithm is given by:

$$\mathbf{u}(k+1) = \mathbf{u}(k) + \frac{1}{k} \mathbf{G}^{-1}(\underline{\sigma}) \mathbf{e}(\mathbf{u}(k)). \quad (29)$$

This algorithm can be motivated either by stochastic approximation theory [7] or as the equivalent recursive version of the least squares solution when the 2-norm of the tracking error is minimised over all iterations up to iteration  $k+1$ . The latter is the same motivation as that used to develop the LPV ILC algorithm so makes the comparison fair. The algorithm is tested in simulation using the same signal for  $\sigma(k)$  and  $y_d(t)$ .

The RMS values  $\|\mathbf{e}(k)\|_2$  achieved using the LPV and LTI ILC algorithms are shown in Figure 1. It can be clearly seen that the proposed method reduces the RMS value of the error substantially over the iterations and the error converges towards zero. There is, however, no obvious trend in the error achieved using the LTI algorithm. This lack of tracking improvement is because the LTI algorithm is not able to learn the correct input as the system's dynamics change continuously between iterations.

#### IV. CONCLUSIONS

An ILC algorithm has been proposed for systems that can be represented by the discrete-time, LPV class of systems. Consistency of the algorithm in the presence of nonstationary stochastic disturbances has been shown when the scheduling parameter trajectory is sufficiently exciting.

The algorithm was tested in simulation and shown to give improved tracking performance over a standard LTI ILC algorithm.

Monotonic convergence of a norm of the error signal is of practical interest, and much attention has been given to this issue in LTI ILC. Unfortunately, it seems unlikely that this property can be incorporated into LPV ILC algorithms as the system's dynamics change from one iteration to the next and so, depending on how they change, it is always possible that the error will increase slightly, though the overall trend should be to decrease.

Furthermore, the proposed method is restricted to LPV systems with a dependence on the scheduling parameter only in the denominator of the system's transfer function. It is clear that a more general algorithm that works for systems with scheduling parameter dependence in the numerator as well would be of interest. A different approach to that presented in this paper would be necessary, nonetheless, as the ideal input could not be represented as a linear combination of the ideal inputs at the polytopic vertices.

#### REFERENCES

- [1] D. Bristow, M. Tharayil, and A. Alleyne, "A survey of iterative learning control," *IEEE Control Systems Magazine*, vol. 26, no. 3, pp. 96–114, June 2006.
- [2] H. S. Ahn, Y. Chen, and K. Moore, "Iterative learning control: Brief survey and categorization," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 6, pp. 1099–1121, Nov. 2007.
- [3] H. Havlicsek and A. Alleyne, "Nonlinear Control of an Electrohydraulic Injection Molding Machine via Iterative Adaptive Learning," *IEEE/ASME Transactions on Mechatronics*, vol. 4, no. 3, pp. 312–323, September 1999.
- [4] D. Wang and Y. Ye, "Design and Experiments of Anticipatory Learning Control: Frequency-Domain Approach," *IEEE/ASME Transactions on Mechatronics*, vol. 10, no. 3, pp. 305–313, June 2005.
- [5] J. Lee and K. Lee, "Iterative learning control applied to batch processes: An overview," *Control Engineering Practice*, vol. 15, no. 10, pp. 1306–1318, 2007.
- [6] A. Tayebi, S. Abdul, M. B. Zaremba, and Y. Ye, "Robust Iterative Learning Control Design: Application to a Robot Manipulator," *IEEE/ASME Transactions on Mechatronics*, vol. 13, no. 5, pp. 608–613, October 2008.
- [7] M. Butcher, A. Karimi, and R. Longchamp, "Iterative learning control based on stochastic approximation," in *17th IFAC World Congress*, Seoul, Korea, July 2008.
- [8] K. Moore and Y. Chen, "A separative high-order framework for monotonic convergent iterative learning controller design," in *IEEE American Control Conference*, Denver, Colorado USA, 2003, pp. 3644–3649 vol.4.
- [9] J. Shamma and M. Athans, "Guaranteed properties of gain scheduled control for linear parameter-varying plants," *Automatica*, vol. 27, no. 3, pp. 559–564, 1991.
- [10] S. Sivrioglu and K. Nonami, "Sliding Mode Control With Time-Varying Hyperplane for AMB Systems," *IEEE/ASME Transactions on Mechatronics*, vol. 3, no. 1, pp. 51–59, March 1998.
- [11] M. da Silva, W. Desmet, and H. V. Brussel, "Design of Mechatronic Systems With Conguration-Dependent Dynamics: Simulation and Optimization," *IEEE/ASME Transactions on Mechatronics*, vol. 13, no. 6, pp. 638–646, December 2008.
- [12] Z. Li, C. Wen, Y. Soh, and Y. Chen, "Iterative learning control of linear parameterized varying uncertain systems," in *6th International Conference on Control, Automation, Robotics and Vision*, Singapore, December 2000.
- [13] H. Stark and J. W. Woods, *Probability, Random Processes and Estimation Theory for Engineers*. New Jersey, U.S.A.: Prentice-Hall, 1986.